COMMUNICATION

7(18), November 1, 2013

ISSN 2278–5469 EISSN 2278–5450

Discovery

The great mechanism: software project management

Shikha Yadav[1], Preeti Dhanda[2], Nisha Yadav[3]

1.Department of Computer Science and Engineering, Dronacharya College of Engineering, Khentawas, Farukhnagar, Gurgaon, India, Email: yadav21shikha@gmail.com
2.Department of Computer Science and Engineering, Dronacharya College of Engineering, Khentawas, Farukhnagar, Gurgaon, India, Email: preeti.dhanda01@gmail.com
3.Department of Computer Science and Engineering, Dronacharya College of Engineering, Khentawas, Farukhnagar, Gurgaon, India, Email: jazzynishu@gmail.com

Publication History
Received: 06 September 2013
Accepted: 20 October 2013
Published: 1 November 2013

Citation
Shikha Yadav, Preeti Dhanda, Nisha Yadav. The great mechanism: software project management. *Discovery*, 2013, 7(18), 13-18

ABSTRACT

The changing environments of software development such as component-based, distributed and subcontracted software development require matching changes by project managers to monitor, control and coordinate their projects. While the objectives of project management may be well established, the mechanisms with which those objectives are achieved are less well known. An empirical study was undertaken to investigate which mechanisms were used by practising project managers to monitor, control and coordinate software development projects. First, the types of mechanisms are discussed so that the mechanisms can be classified usefully. Then, the design of the empirical study is described. The data were composed through structured interview, which provided both quantitative and qualitative data. The data are analysed for each mechanism separately and the outcomes presented. The study found that project managers use multiple mechanisms to achieve project management objectives and use the same mechanism to serve multiple objectives. Further research is suggested to investigate project management from the contrary orientation, that is, which objectives are served by Specific project management mechanisms.

Keywords: software; projects; data

1. INTRODUCTION

As software development projects adapt to changing circumstances, management of those projects must also adapt. Such changes could involve adopting different development life cycles, moving to component-based software development or distributing the development. While the project management objectives may remain the same, the mechanism used to achieve those objectives will not essentially remain unchanged and adaptation would be assisted through understanding which mechanisms are used in different circumstances. Such knowledge of the mechanisms could guide the ways in which project managers adapt to different project environments as well as guide efforts to develop project workflow and project management tools.

This research investigates the mechanisms of project monitoring, control and coordination during the project's development phase. It concentrates on the development phase because that is when the project is subject to the unexpected events and external influences that require the project manager to take action to keep the project moving toward its objectives. It does not investigate project planning or any other activity that might be undertaken by the project manager prior to the start of actual software development because project management during that phase is concerned with producing the project plan, not developing the software. Nor does the research investigate activities undertaken after the software's development such as deployment or support and maintenance for similar reasons. Concentrating on the development phase means that project managers of all projects are likely to consider the same range of project management mechanisms, the choice of which is likely to be subject to similar range of influences.

2. HISTORY

In the 1970s and 1980s, the software industry grew very rapidly, as computer companies quickly renowned the relatively low cost of software production compared to hardware production and circuitry. To manage new development efforts, companies applied the reputable project management methods, but project schedules slipped during test runs, especially when confusion occurred in the gray zone between the user specifications and the delivered software. To be able to avoid these problems, *software* project management methods focused on matching user requirements to delivered products, in a method known now as the Waterfall Model. As the industry has matured, study of software project management failures has shown that the following are the most common causes:

- Unrealistic or unspoken project goals
- Inaccurate estimates of needed resources
- Poorly defined system requirements
- reduced reporting of the project's status
- Unmanaged risks
- Poor communication among customers, developers, and users
- employ of immature technology
- Inability to handle the project's complexity
- Poor project management

The list above shows the hitches articulating the requirements of the client in such a way that proper resources can deliver the proper project goals. Specific *Software Project Management Tools* are helpful and often essential, but the true art in software project management is applying the correct method and then using tools to hold up the method. Without a method, tools are insignificant. Since the 1960s, several proprietary software project management methods have been developed by software manufacturers for their own use, while computer consulting firms have also developed parallel methods for their clients. Today software project management methods are still evolving, but the present trend leads away from the waterfall model to a more cyclic project delivery model that imitates a Software release life cycle.

3. SOFTWARE DEVELOPMENT PROCESS

A Software Development Process is concerned principally with the production aspect of Software Development, as contrasting to the technical feature, such as Software Tools. These processes survive primarily for supporting the management of software development, and are generally skewed toward addressing business concerns. Many software development processes can be run in a similar way to general project management processes. Examples are:

- **Risk Management**: it is the process of measuring or *assessing risk* and then developing strategies to supervise the risk. In general, the strategies employed include transferring the risk to another party, avoiding the risk, reducing the unhelpful result of the risk, and accepting some or all of the consequences of a particular risk. Risk management in software project management

begins with the business case for opening the project, which includes a cost beneficial analysis as well as a list of fallback options for project failure, called a contingency plan.

- **Requirement Management**: it is the course of identifying, eliciting, documenting, analyzing, tracing, prioritizing and harmonizing on requirements and then controlling transform and communicating to relevant stakeholders. Fresh or altered *Computer System* Requirements management, which includes Requirement analysis, is an significant part of the Software Engineering process; whereby business analysts or software developers spot the requirements of a client; having recognized these requirements they are then in a position to plan a solution.
- **Change Management**: it is the procedure of identifying, documenting, analyzing, prioritizing and agreeing on changes to scope (Project Management) and then controlling changes and communicating to significant stakeholders. Change impact analysis of new-fangled or altered scope, which includes Requirement analysis at the change level, is an significant part of the software engineering process; whereby business analysts or software developers recognize the transformed needs or requirements of a client; having known these requirements they are then in a position to re-design or alter a solution. Hypothetically, each change can impact the timeline and budget of a software project, and therefore by definition must include risk-benefit analysis before approval.
- **Software Configuration Management**: it is the course of identifying, and documenting the scope itself, which is the software product ongoing, together with all sub-products and changes and enabling communication of these to relevant stakeholders. In wide-ranging, the processes in employment including version control, naming convention (programming) and software archival agreements.
- **Release Management**: it is the course of identifying, documenting, prioritizing and harmonizing on releases of software and then controlling the release agenda and communicating to relevant stakeholders. Most software projects have admission to three software environments to which software can be released; Development, Test, and Production. In very large projects, where distributed teams necessitate integrating their effort before releasing to users, there will often be more environments for testing, called Unit testing, System testing or Integration testing before release to User acceptance testing (UAT).
- A subset of release management that is getting hold of more and more concentration is Data Management, as visibly the users can only test based on data that they know, and "real" data is only in the software environment called "production". In order to test their effort, programmers must therefore also often create "dummy data" or "data stubs". Conventionally, older versions of a production system were once used for this purpose, but as companies rely more and more on outside contributors for software development, company data may not be released to development teams. In complex environments, datasets may be created that are then migrated across test environments according to a test release schedule, much like the overall software release schedule.

4. PROJECT MONITORING

Project monitoring is the gathering of information to determine the current state and progress of the project in relation to its expected state and progress. Often, a project monitoring support project control and is so frequently associated with project control that the two are treated as inseparable. Yet project monitoring and project control are not the same activity, nor are they inseparable. Control includes directing efforts toward some end objectives and sometimes considers information gathering to be part of the control activities. While this is a useful view when studying organizational control, it excludes information gathering for other purposes such as quality assurance or coordination. In this discussion, it is useful to separate monitoring from both control and coordination so that monitoring mechanisms can be identified independently of the intended use of the information, and so that the effect of organizational distance on monitoring can be considered independently. A search of the literature revealed that monitoring mechanisms seem to fall into four groupings:

- **Automatic monitoring:** Information that can be gathered automatically from Software development or project management tools and systems.
- **Formal monitoring:** Information that is gathered through formal administrative systems.
- **Ad hoc monitoring:** Information gathered through irregular enquiry such as audits and reviews.
- **Informal monitoring:** Information gathered informally through conversations or their equivalent.

5. PROJECT COORDINATION

Coordination is concerned with managing the interdependencies between activities or among multiple individuals involved in an activity. The purpose of coordination is to "coordinate the work so that it gets done and fits together, so that it is not done redundantly, and so that the components of the work are handed off expeditiously." The most frequently cited definition of

coordination is that of *Malone and Crowston (1994)* in which coordination is defined as "*managing the dependencies between activities*". Their analysis considers shared resources, producer/consumer relationships, simultaneity constraints and task/subtask dependencies. Thus, their model considers the relationships between actors, actions, the acted upon and the constraints that may operate on combinations of all three. Sabherwal (2003) examined prior literature to arrive at a broad classification of coordination mechanisms into plan, standard, formal mutual adjustment and informal mutual adjustment. The mechanisms identified by Sabherwal are distinguished by their fixed and variable costs. The more formal coordination mechanisms have higher initial costs but lower variable costs. The less formal mechanisms have a lower initial cost but higher variable costs as illustrated in figure 1. The available models of coordination all display some form of continuum from more formal and less interactive to less formal and more interactive. Since this research needs to identify the different coordination mechanisms rather than deal with an undifferentiated group of mechanisms, it will adopt the classifications proposed by Sabherwal as a result of considering the body of research concerning coordination mechanisms; that is, plan-based, standards-based, formal mutual adjustment and informal mutual adjustment.

- **Coordination by standards**. The distinguishing feature of this type of mechanism is that it is concerned with the rules by which something is done rather than the goals that guide the development.
- **Coordination by plans**. When coordination is achieved by specifying what is to be produced and, possibly, when it is to be produced, then this can be described as coordination by plans.
- **Coordination by formal mutual adjustment**. Overcoming some problems requires some form of mutual adjustment, some of which can be formalized into specific actions or mechanisms. Kraut and Streeter (1995) offer modification request tracking and data dictionaries as examples of formal mutual adjustment. Additionally lists design review meetings, reporting requirements and liaison roles among the formal mutual adjustment mechanisms.
- **Coordination by informal mutual adjustment**. Informal mutual adjustment is the most flexible all the types of coordination mechanisms but comes with a high variable cost.
-

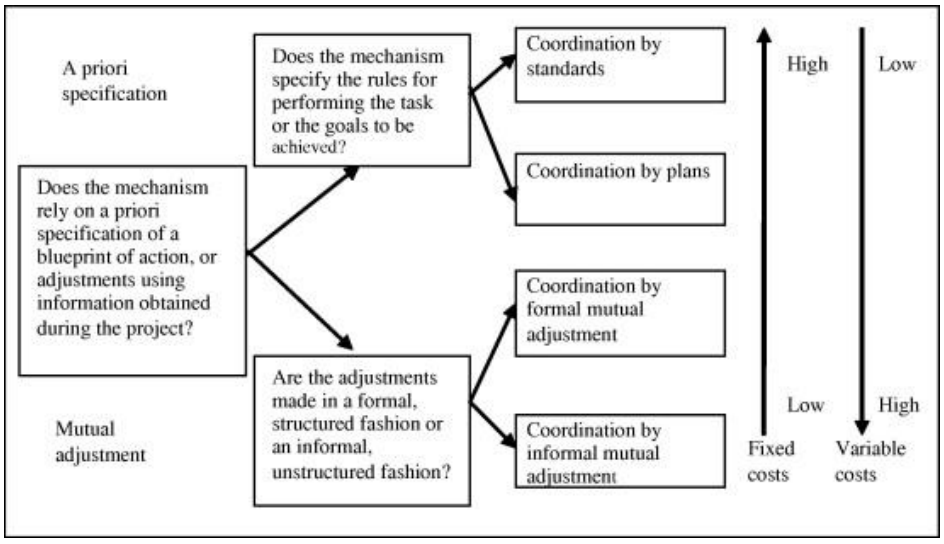

Figure 1

The classification of coordination mechanisms - from Sabherwal (2003)

6. SIGNIFICANCE TO PROJECT MANAGERS

This research has recognized that a range of mechanisms are available to project managers and has shown the reasonable popularity of each. Although the research has limited external validity due to the relatively small number of interviews, it is believed to be reasonably representative of the software development industry. The research suggests that not only are there many mechanisms to monitor, control and coordinate software development projects but that numerous of them might be employed on the same project. Certainly there was some indication that the more experienced project managers employed several mechanism to increase the richness of information available to them and to verify the consistency of the information. The research did not extend

to identifying which circumstances or project types tended to favour specific mechanisms. Instead this research simply seeks to identify current practice.

7. DISCUSSING VARIOUS ASPECTS OF SOFTWARE PROJECT MANAGEMENT

This research investigated the use of project management mechanisms by considering each type of mechanism (monitor, control and coordinate) discretely then later observing that project managers frequently use the same mechanism for multiple purposes. This suggests that the opposite approach could be fruitful, i.e., to first establish which mechanisms project managers use and the different purposes for which they are used. A project management mechanism could be used for different objectives, depending on the circumstances. For example, a meeting may be used to enforce project control and, later, another meeting be used to clarify how some parts of the system must interact, which is a coordination matter. More likely is that mechanisms will be used for several purposes at the same time. A mechanism may be used to achieve a primary objective as well as a secondary objective. For example, a specification such as a system design is primarily a control mechanism since it specifies what is to be achieved. However, the same specification acts as a coordination mechanism by specifying the how different parts of the system will interact. The primary, secondary or incidental purposes for which a mechanism is used could vary according to conditions and this would need to be investigated.

Understanding how the different mechanisms are used would reveal more about how project managers actually monitor, control and coordinate their projects and lead to a better understanding of requirements for tools to assist them. Further research could also be undertaken to investigate the effectiveness of each mechanism or the reasons for their choice. While some mechanisms might be preferred in more formal software development projects there is little data available on the benefits of the mechanism compared to the cost of employing it.

Project monitoring Project managers favour formal and informal monitoring mechanisms over automatic or ad hoc monitoring mechanisms. The most familiar monitoring mechanism is the weekly project review meeting with the development team that reviews the schedule and milestones as well as other project issues. There is little evidence of adopting such automated tools as workflow systems but some evidence that iterative development is being used with its frequent release cycles permitting a form of automatic monitoring. Informal monitoring through conversations with the project team and with the customer is widely practised. One of the unexpected findings is that many project managers that use schedule and milestone tracking to monitor the general state of the project but supplement this with ad hoc “drill down” enquiries whenever there are any slippages or even threats of slippages. Such an Early Warning system of monitoring would appear to be an inexpensive and unobtrusive way to monitor the project and react quickly when needed.

Early warning systems

The analysis clearly showed that project managers used the formal project monitoring mechanisms as a warning system to indicate that something needs further investigation. If any alarm was triggered in the mind of the project manager then ad hoc monitoring was initiated in the form of a specific inquiry or something more formal such as a project audit. This is an efficient way to monitor any project. If the formal monitoring does not need to gather a lot of information then it can be pared down to only that which indicates the health of the project rather than having to include information that would indicate all the possible causes of ill health. Causes of “ill health” can be sought through a more directed inquiry that needs only involve those directly concerned with the particular problem and not the entire project team.

Multiple sources of information

Project managers tended to use multiple sources of information about the same aspect of the project. Mostly this was the project’s progress, which could be measured reasonably clearly, but also included less measurable attributes, such as the project’s health. This indicates that multiple sources of information will be sought to overcome information uncertainty. If the data cannot be readily verified by the project manager, then multiple sources of the same project attribute will triangulate the information, thus reducing uncertainty. If the project or product attribute is difficult to measure, then multiple related measures will tend to reduce the inaccuracy of single measures. For example, it is difficult to monitor a single project characteristic that would indicate the health of the project and the probability that it will be completed on schedule. Instead, several measures are sought that can collectively indicate the project’s health. This research did not pursue this apparent relationship between information uncertainty and the number of information sources. That will be left for further research.

8. SCOPE

This research investigated the use of project management mechanisms by considering each type of mechanism (monitor, control and coordinate) separately then later observing that project managers frequently use the same mechanism for multiple purposes. This suggests that the opposite approach could be fruitful, i.e., to first establish which mechanisms project managers use and the different purposes for which they are used. There has not been a rigorous analysis nor any empirical data on how any one mechanism is used by a project manager to achieve multiple objectives.

A project management mechanism could be used for different objectives, depending on the circumstances. For example, a meeting may be used to enforce project control and, later, another meeting be used to clarify how some parts of the system must interact, which is a coordination matter. More likely is that mechanisms will be used for several purposes at the same time. A mechanism may be used to attain a primary objective as well as a secondary objective. For example, a specification such as a system design is primarily a control mechanism since it specifies what is to be achieved. However, the same specification acts as a coordination mechanism by specifying the how different parts of the system will interact. The primary, secondary or incidental purposes for which a mechanism is used could vary according to circumstances and this would need to be investigated. Understanding how the different mechanisms are used would reveal more about how project managers actually monitor, control and coordinate their projects and lead to a better understanding of requirements for tools to assist them.

Further research could also be undertaken to investigate the effectiveness of each mechanism or the reasons for their choice. While some mechanisms might be favoured in more formal software development projects there is little data available on the benefits of the mechanism compared to the cost of employing it.

REFERENCE

1. Addison T, Vallabh S. Controlling software project risks: an empirical study of methods used by experienced project managers, *Proceedings of the 2002 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology, Port Elizabeth*, South African Institute for Computer Scientists and Information Technologists, 2002, pp. 128-140
2. Jalote, Pankaj. *Software project management in practice*. Addison-Wesley. 2002, ISBN 0-201-73721-3
3. John C Reynolds. *Some thoughts on teaching programming and programming languages*, SIGPLAN Notices, 2008, 43(11), p.108: "Some argue that one can manage software production without the ability to program. This belief seems to arise from the mistaken view that software production is a form of manufacturing. But manufacturing is the repeated construction of identical objects, while software production is the construction of unique objects, i.e., the entire process is a form of design. As such it is closer to the production of a newspaper [sic] — so that a software manager who cannot program is akin to a managing editor who cannot write."
4. Murali Chemuturi, Thomas M Cagley Jr. *Software Project Management: Best Practices, Tools and Techniques*. J.Ross Publishing. 2010, ISBN 978-1-60427-034-1
5. Stellman Andrew, Greene Jennifer. Applied Software Project Management. O'Reilly Media. 2005, ISBN 978-0-596-00948-9